

What are the productive units of natural language grammar?

Willem Zuidema
Institute for Logic, Language and Computation
University of Amsterdam

Outline

- Linguistic Motivation
- Data-Oriented Parsing as Probabilistic Construction Grammar
- Identifying constructions = learning a grammar
 - Problems with existing DOP estimators
 - New estimator: push-n-pull
 - Results

Probabilistic Linguistics

(Abney 1996, *Statistical Methods and Linguistics*)

- (1) a. #the a are of I word salad?
- b. John saw Mary unambiguous?

- (2) a. a hectare is a hundred ares
- b. As described in section I paragraph a ...
- c. The a paragraph of I is hardly readable.
- d. Typhoid Mary
- e. the Russia house butler

Constructions

(Fillmore, Kay & O'Connor, 1988; Culicover & Nowak, 2004; Jackendoff, forthcoming)

Idioms

- (3) a. by and large
- b. lo and behold
- c. beat a dead horse
- d. make amends
- e. cast aspersions
- f. a flash in the pan

Constructions

Idiosyncratic syntactic behavior

- (4) a. Pat sang/drank/sewed his heart out
- b. *Pat sang the Marseillaise his heart out

- (5) a. Elmer hobbled/laughed/joked his way to the bank.
- b. Hermione slept/drank/sewed/programmed three whole evenings away.

Constructions

Syntax-semantics mismatches

- (6) a. eat a quick/occasional/leisurely bagel
- b. burned a *quick/occasional bagel

Formulaic language, non-head dependencies

- (7) a. When is the next train *from Amsterdam to* Paris?
- b. BA carried *more people than* cargo in 1987.
- c. Lawson is *the closest thing* in London *to* a supply-side globalist.

Radical Construction Grammar

(Croft, Jackendoff)

- Constructions map conceptual, syntactic and phonological structures onto each other;
- At each level, there can be a lot, very little or nothing specified;
- Each construction has an associated weight (frequency/recency information);
- Everything is a construction, from specific words to highly abstract word order principles.

Stochastic Tree Substitution Grammars

<div style="display: flex; justify-content: space-around; align-items: flex-start;"> <div style="text-align: center;"> <p>NP</p> <p>PP-DIR PP-DIR</p> <p>.2</p> </div> <div style="text-align: center;"> <p>PP-DIR</p> <p>IN NNP</p> <p>from .5</p> </div> <div style="text-align: center;"> <p>NNP</p> <p>Baltimore</p> <p>.5</p> </div> <div style="text-align: center;"> <p>PP-DIR</p> <p>TO NNP</p> <p>to .5</p> </div> <div style="text-align: center;"> <p>NNP</p> <p>Oakland</p> <p>.5</p> </div> </div>	$\Pi = 0.0125$
<div style="display: flex; justify-content: space-around; align-items: flex-start;"> <div style="text-align: center;"> <p>NP</p> <p>PP-DIR PP-DIR</p> <p>IN NNP TO NNP</p> <p>from to .2</p> </div> <div style="text-align: center;"> <p>NNP</p> <p>Baltimore</p> <p>.5</p> </div> <div style="text-align: center;"> <p>NNP</p> <p>Oakland</p> <p>.5</p> </div> </div>	$\Pi = 0.05$
<div style="display: flex; justify-content: space-around; align-items: flex-start;"> <div style="text-align: center;"> <p>NP</p> <p>PP-DIR PP-DIR</p> <p>IN NNP TO NNP</p> <p>from Baltimore to Oakland .1</p> </div> </div>	$\Pi = 0.1$ $\Sigma = 0.1625$

Stochastic Tree Substitution Grammars

An STSG is a 5-tuple $\langle V_n, V_t, S, T, w \rangle$

$$w : T \rightarrow [0, 1], \text{ such that } \forall r \sum_{t:r(t)=r} w(t) = 1$$

The probability of a derivation:

[hidden]

$$P(d = t_1 \circ \dots \circ t_n) = \prod_{i=1}^n (w(t_i))$$

The probability of a parse:

[observable]

$$P(p) = \sum_{d:\hat{d}=p} (P(d))$$

Stochastic Tree Substitution Grammars

Expected Usage Frequency:

[hidden]

$$u(t) = \sum_{d:t \in d} P(d) C(t, d)$$

Expected Occurrence Frequency:

[observable]

$$\mathbf{E}[f(t)] = \sum_{p:t \in p^*} P(p) C(t, p^*),$$

(where p^* is set of all subtrees of p , and $C(t, p^*) = \#$ occurrences of t in p^*)

Data-Oriented Parsing

(Scha, 1990; Bod, 1993)

1. Symbolic grammar is based on fragments observed in the tree bank (“tree bank grammar”);
2. Subtrees of arbitrary size are considered as elementary units;
3. Occurrence and co-occurrence frequencies of subtrees are basis for disambiguation.

Data-Oriented Parsing

DOP1 (Bod, 1993, 1998)

$$w(t) = \frac{f(t)}{\sum_{t':r(t')=r(t)} f(t')}$$

L-DOP (Goodman, 2003; Bod, 2003)

$$w(t) = \alpha_t \frac{f(t)}{\sum_{t':r(t')=r(t)} f(t')}$$

(where α_t is a scaling factor)

Maximum Probable Parse

Excellent empirical results

E.g. on Wall Street Journal sentences of less than 100 words:

parser	LP	LR	<i>F</i>
Collins '96	.857	.853	.855
Collins '99	.883	.881	.882
Charniak '00	.895	.896	.895
L-DOP	.897	.895	.896
SL-DOP	.908	.907	.907
Charniak & Johnson '05			.910
this conference			~.920

LP=labeled precision, LR=labeled recall, $F = 2 \cdot LP \cdot LR / (LP + LR)$

Computational problems

- Time complexity approximations of MPP
- Space complexity PCFG reduction
- Inconsistency of Estimators (EACL'06)
- Data Annotation U-DOP

What are the relevant constructions?

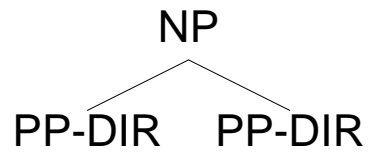
TO
|
to
1.0

CC
|
and
46/54

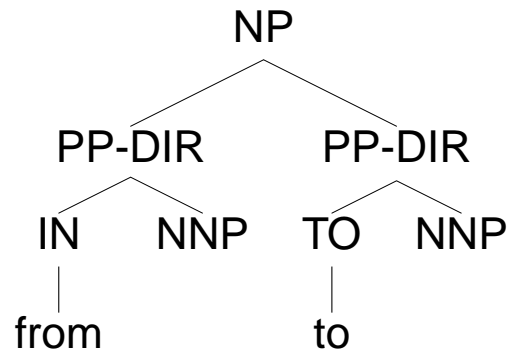
What are the relevant constructions?

TO
|
to
1.0

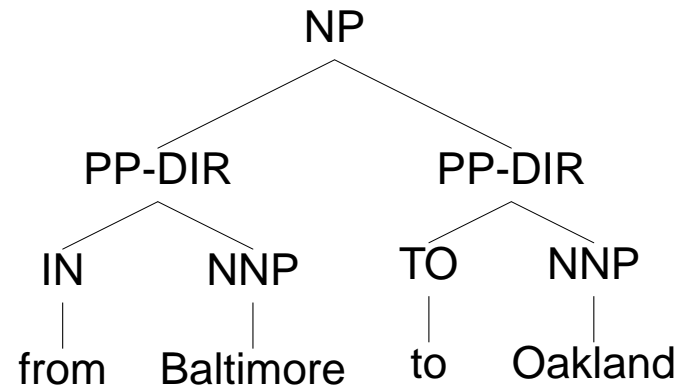
CC
|
and
46/54



$w_1 \geq$



$w_2 \geq$

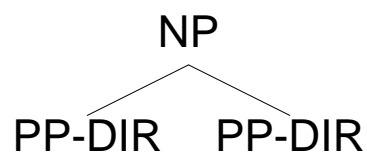


w_3

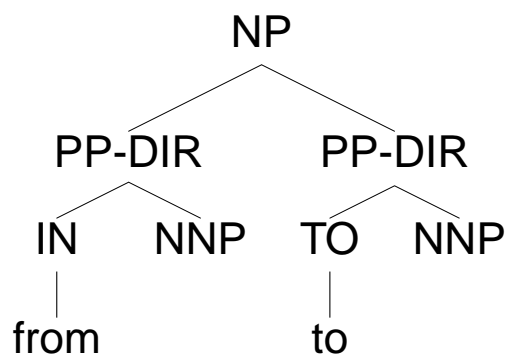
What are the relevant constructions?

TO
|
to
1.0

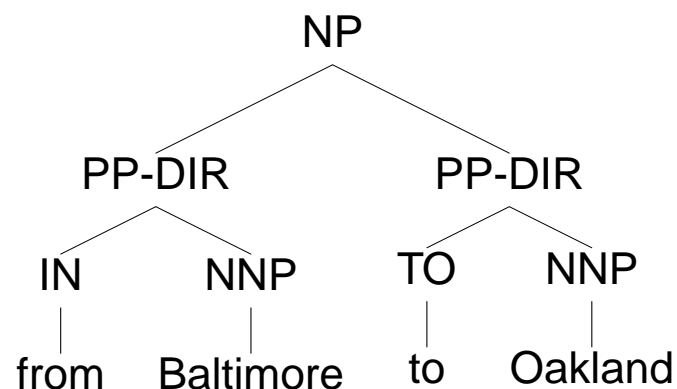
CC
|
and
46/54



$w_1 \geq$



$w_2 \geq$



w_3

Bod (2003) ("Do All Fragments Count?", Natural Language Engineering, 9(4), 307-323) finds that any across-the-board restriction on the allowed fragments, decreases parse accuracy.

- DOP1 (Bod, 1993) and related estimators give observed fragments of all sizes non-zero weights, regardless of whether their frequency is already explained by smaller fragments;
- DOP* (Zollmann & Sima'an, 2005), S-DOP (Bod, 2003) and related estimators push all probability mass to the largest fragments, regardless of whether their frequency can be explained by smaller fragments equally well.

(Zuidema, *EACL*, 2006)

A new estimator: push-n-pull

- For every fragment t , calculate the difference between observed and expected frequency: $\Delta_1 = \mathbf{E}[f(t)] - f(t)$;
- If $\Delta_1 > 0$, the current grammar overestimates the frequency of t . Hence, as far as possible, *push* some of its weight to other trees.
- If $\Delta_1 < 0$, the current grammar underestimates the frequency of t . Hence, as far as possible, *pull* some weight from other trees.

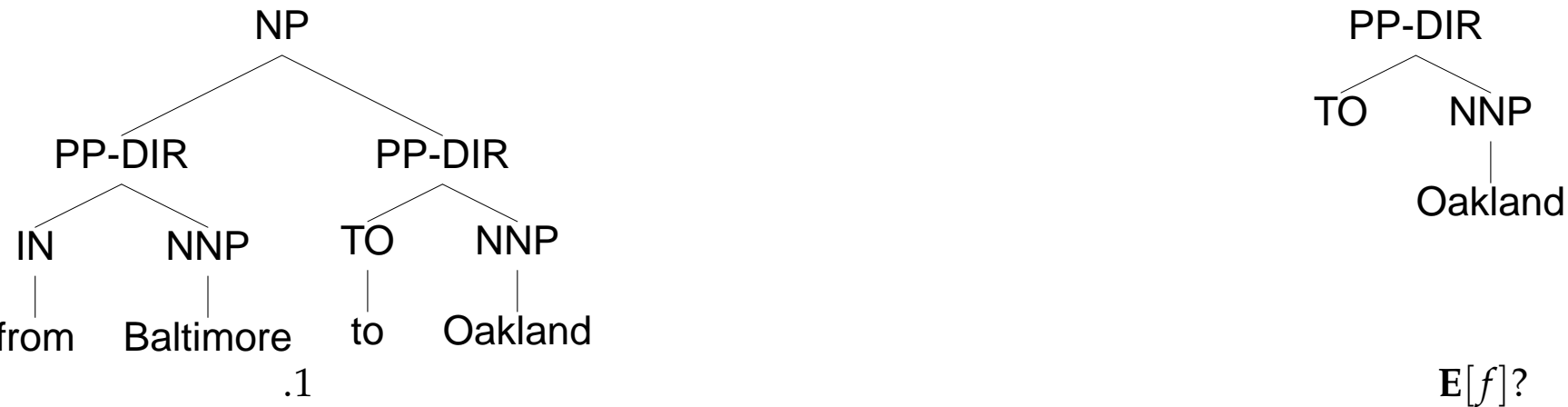
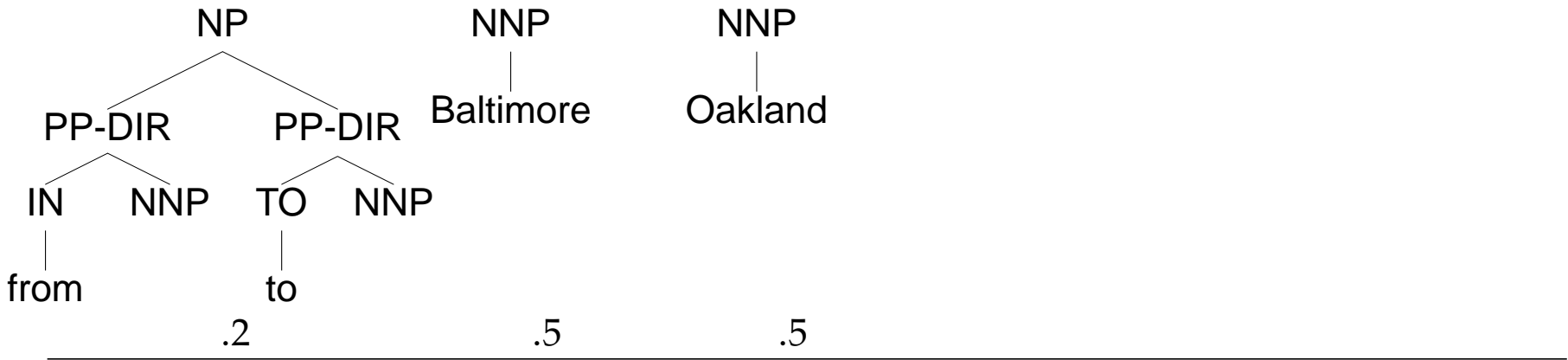
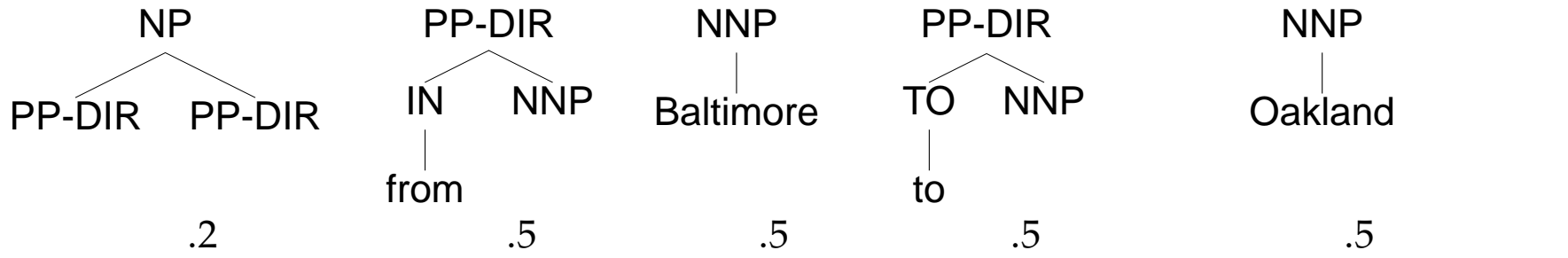
Expected Frequency

$$\mathbf{E}[f(t)] = \sum_{d \in D(t)} (\alpha(d)\beta(d))$$

$$\alpha(d) = \sum_{\tau \in \widehat{tw}(d_1)} \left(\sum_{\tau' \in \widehat{pr}_{x(t)}(\tau)} u(\tau') \right)$$

$$\beta(d) = \prod_{\substack{t' \in \\ \langle d_2, \dots, d_n \rangle}} \left(\sum_{\tau' \in \widehat{pr}_{x(t)}(t')} w(\tau') \right)$$

Expected Frequency



Push

- Pushing scores away is possible upto the current score of t . Hence, $\Delta' = \text{MINIMUM}(sc(t), \Delta)$;
- The score of t is decreased: $sc(t)_- = \Delta'$;
- The score is *pushed* towards all trees t' , which are subtrees of t involved in length-2 derivations of t . Hence: $sc(t')_+ = \Delta' / \#\text{such derivations}$

Pull

- Scores are pulled from all trees t' , which are subtrees of t involved in length-2 derivations of t .
- Pulling scores is only possible upto the point where these subtrees have score 0. If there are n length-2 derivations d^i of t , then $\delta^i = \text{MINIMUM}(sc(d_1^i), sc(d_2^i), -\Delta/n)$.
- The scores of these subtrees are decreased: $sc(d_j^i)_- = \delta_i$;
- The score of t is increased: $(t)_+ = \sum_i \delta_i$

for each observed parse tree p
 for each depth-1 subtree t in p
 update-score($t, 1.0$)

for each observed parse tree p
 for each depth-1 subtree t in p
 update-score($t, 1.0$)
 for each subtree t of p
 $\Delta = \min(sc(t), B + \gamma(\mathbf{E}[f(t)] - f(t)))$

for each observed parse tree p
 for each depth-1 subtree t in p
 update-score($t, 1.0$)
 for each subtree t of p
 $\Delta = \min(sc(t), B + \gamma(\mathbf{E}[f(t)] - f(t)))$
 $\Delta' = 0$
 for each of n derivations d of t
 let $t' \dots t''$ be all elementary trees in d
 $\delta = \min(sc(t'), \dots, sc(t''), -\Delta/n)$

for each observed parse tree p
 for each depth-1 subtree t in p
 update-score($t, 1.0$)
 for each subtree t of p
 $\Delta = \min(sc(t), B + \gamma(\mathbf{E}[f(t)] - f(t)))$
 $\Delta' = 0$
 for each of n derivations d of t
 let $t' \dots t''$ be all elementary trees in d
 $\delta = \min(sc(t'), \dots, sc(t''), -\Delta/n)$
 $\Delta' - = \delta$
 for each elementary tree t' in d
 update-score(t', δ)
 update-score(t, Δ')

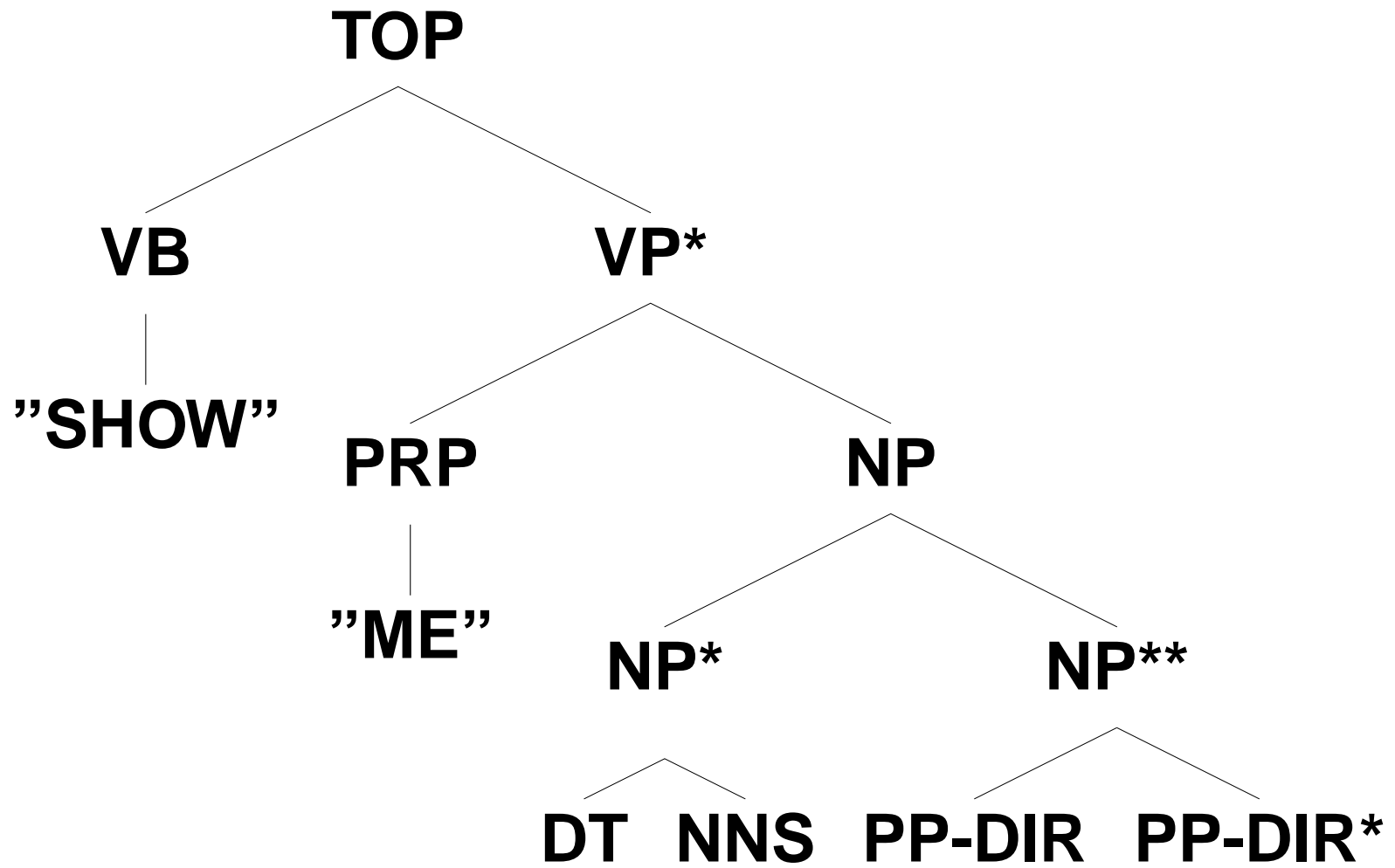
for each observed parse tree p
 for each depth-1 subtree t in p
 update-score($t, 1.0$)
 for each subtree t of p
 $\Delta = \min(sc(t), B + \gamma(\mathbf{E}[f(t)] - f(t)))$
 $\Delta' = 0$
 for each of n derivations d of t
 let $t' \dots t''$ be all elementary trees in d
 $\delta = \min(sc(t'), \dots, sc(t''), -\Delta/n)$
 $\Delta' = \delta$
 for each elementary tree t' in d
 update-score(t', δ)
 update-score(t, Δ')

Results

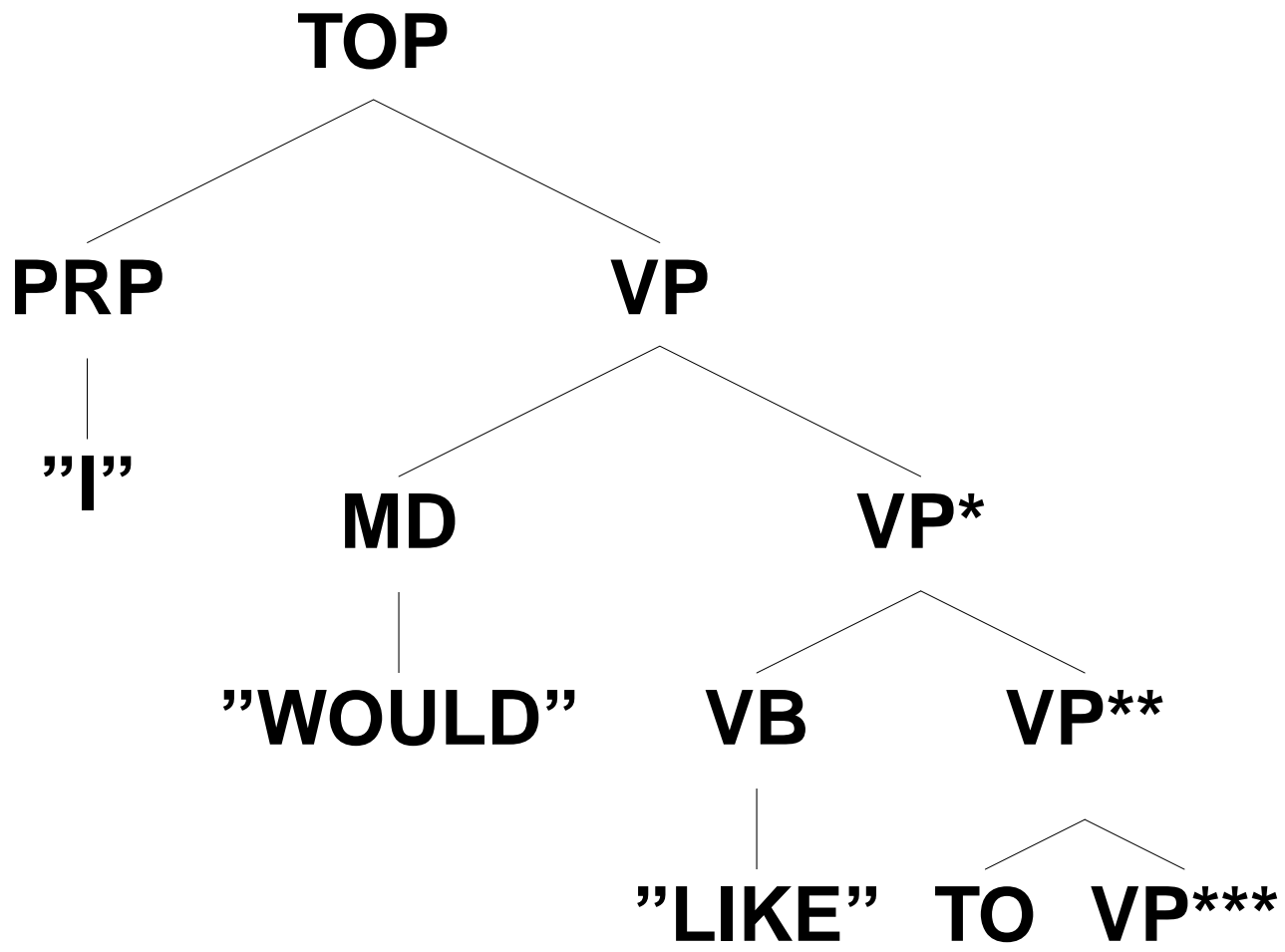
Evalb scores using a random 462-116 trainset-testset split of ATIS3, converted to CNF with unique nonterminal labels (and “I’d” replaced by “I would”):

method	# rules	Cov.	LR	LP	EM
DOP1	77852	84%	95.07	95.07	83.5
p-n-p	58799	84%	95.07	95.07	83.5

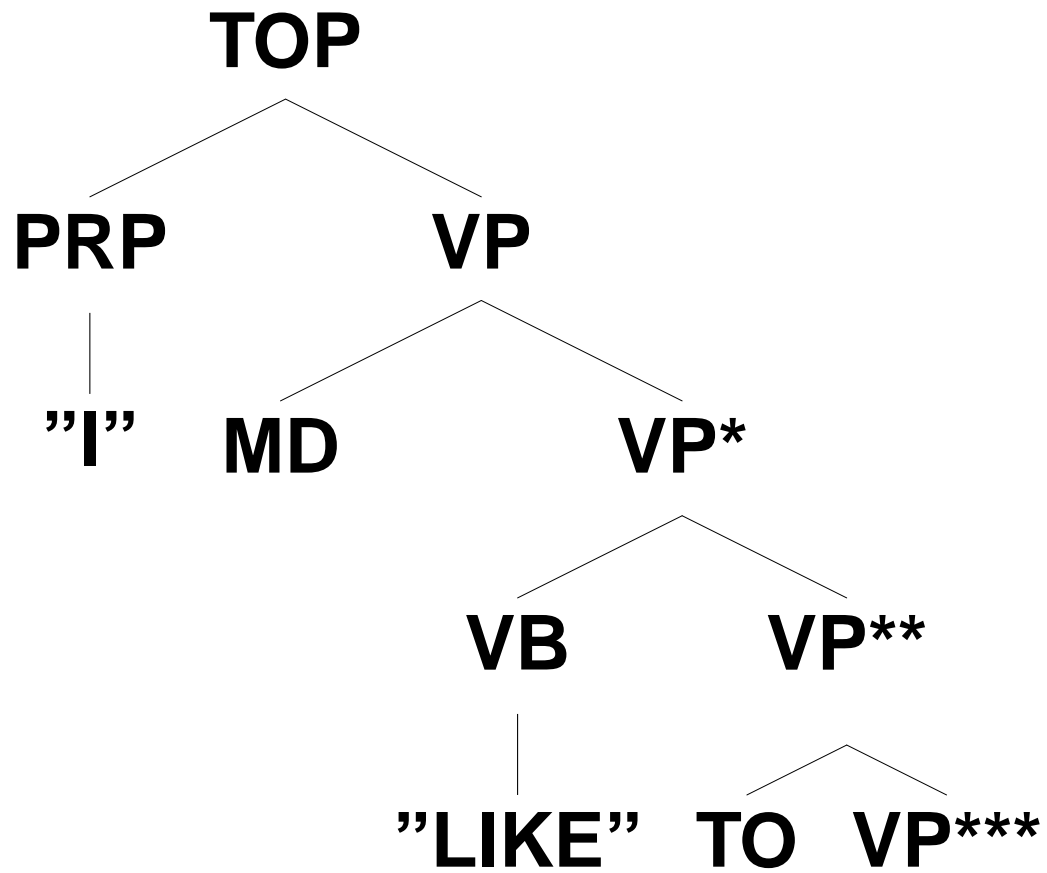
(Experiments on Penn WSJ corpus will follow soon).



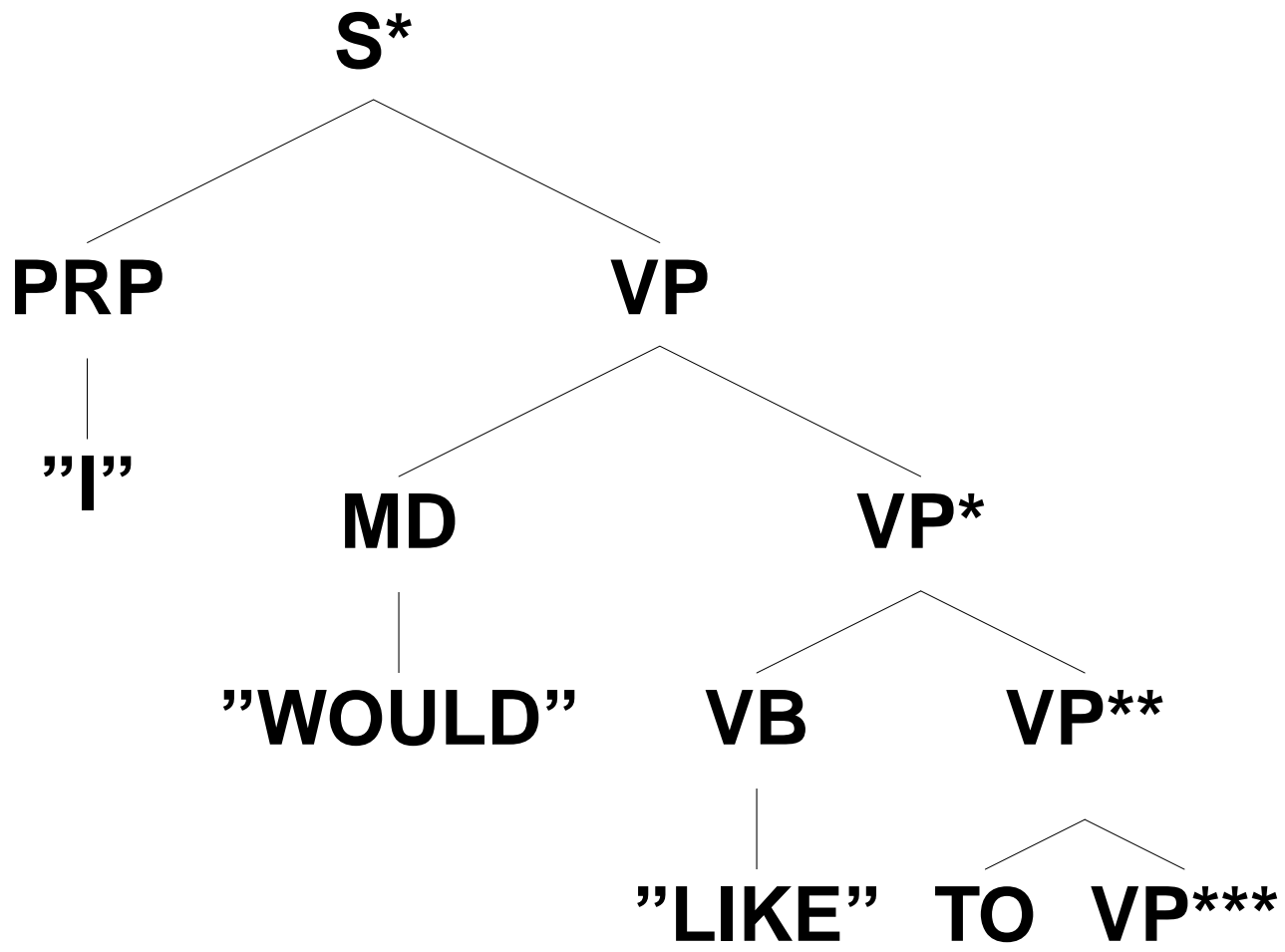
17.79 0.008 30



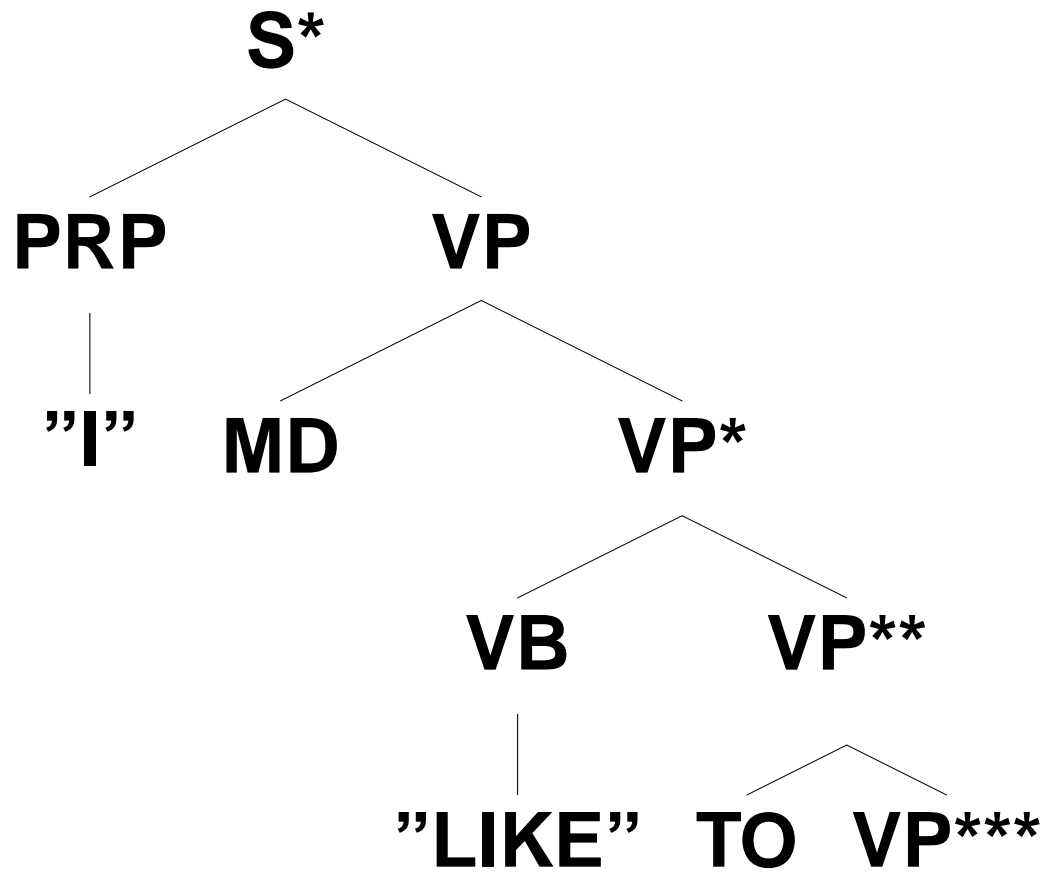
10.02 0.009 20



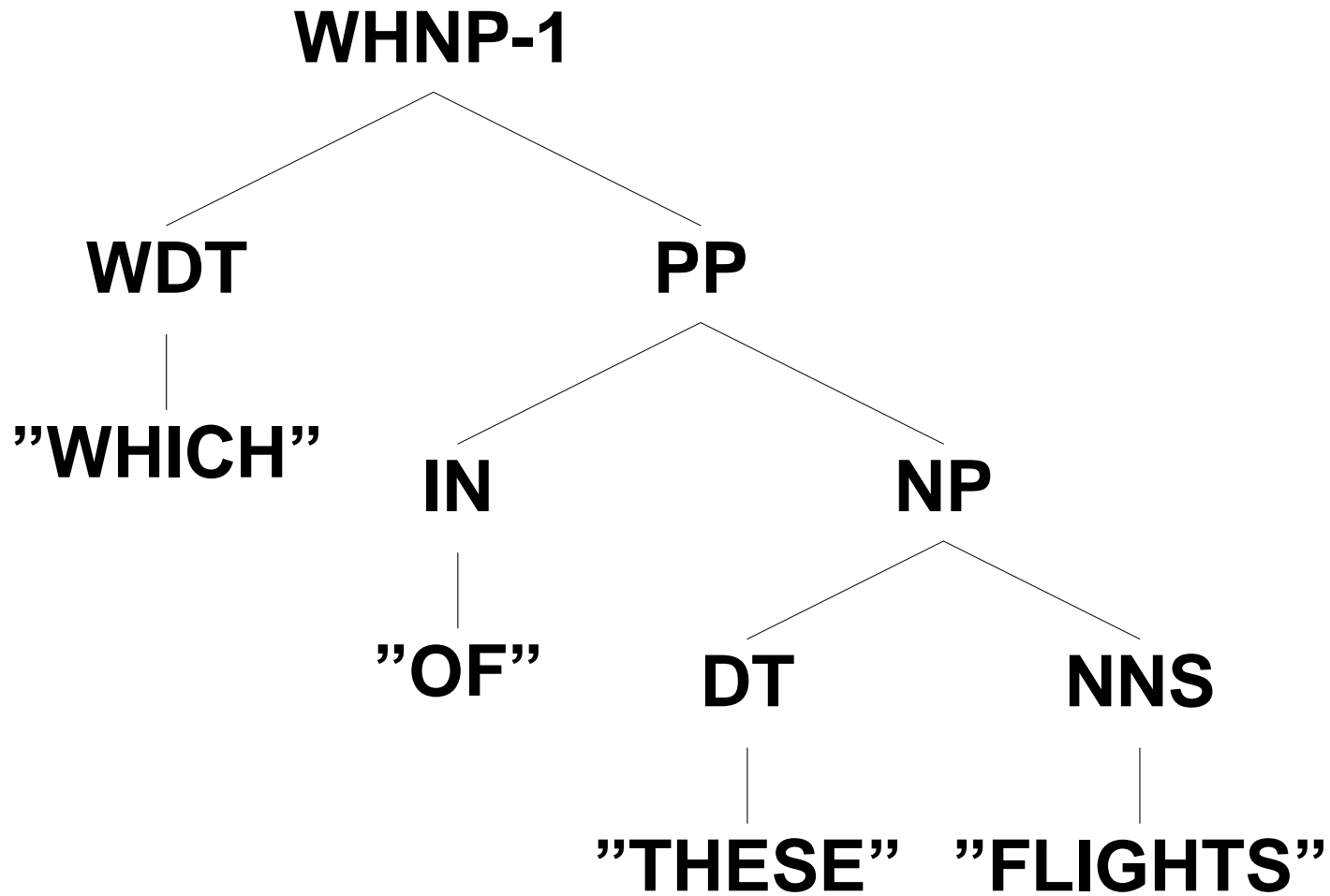
6.48 0.006 20



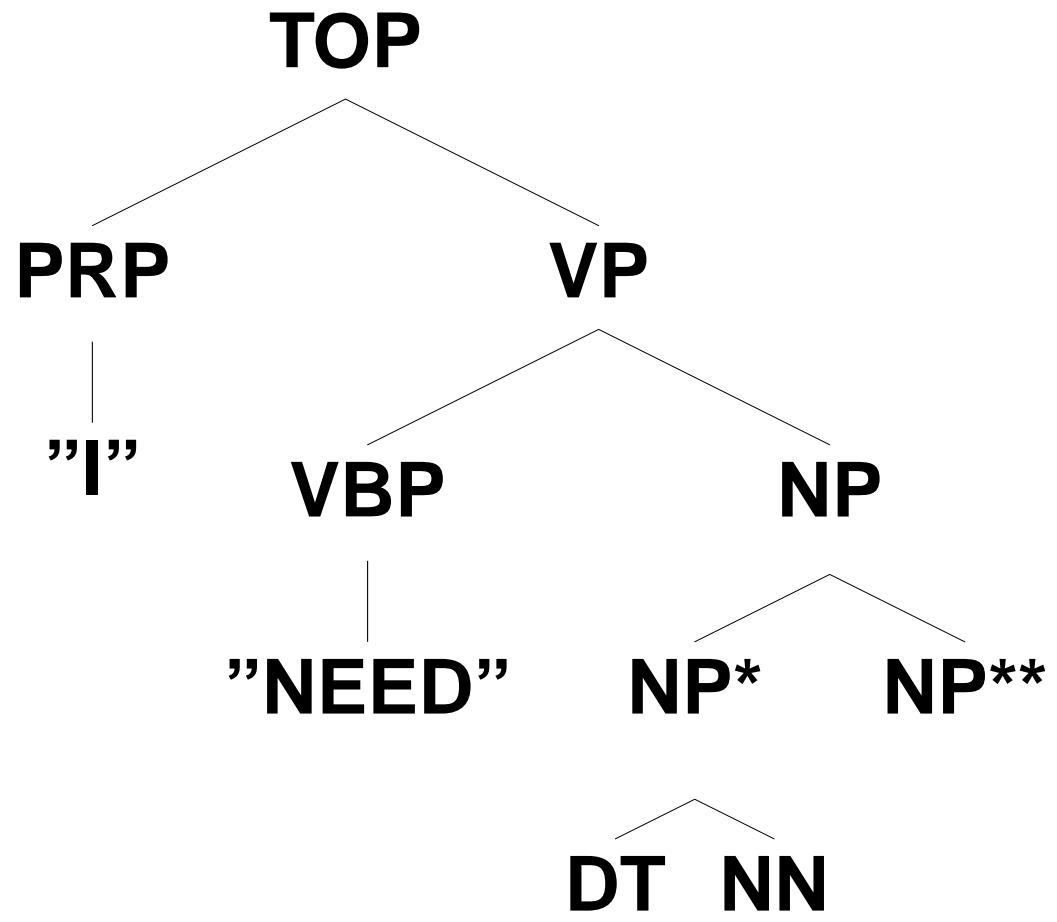
2.59 0.11 8



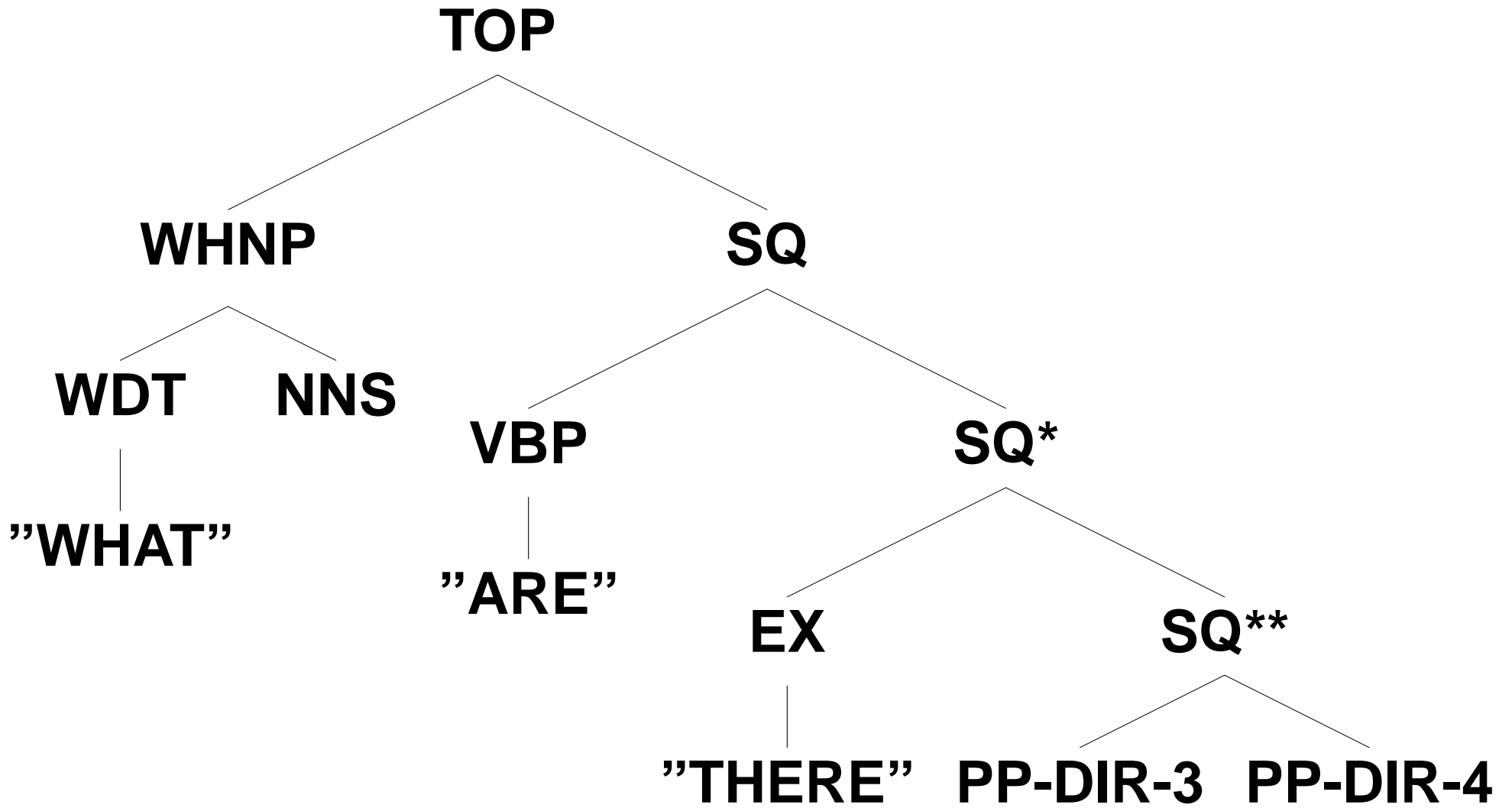
2.33 0.100 8



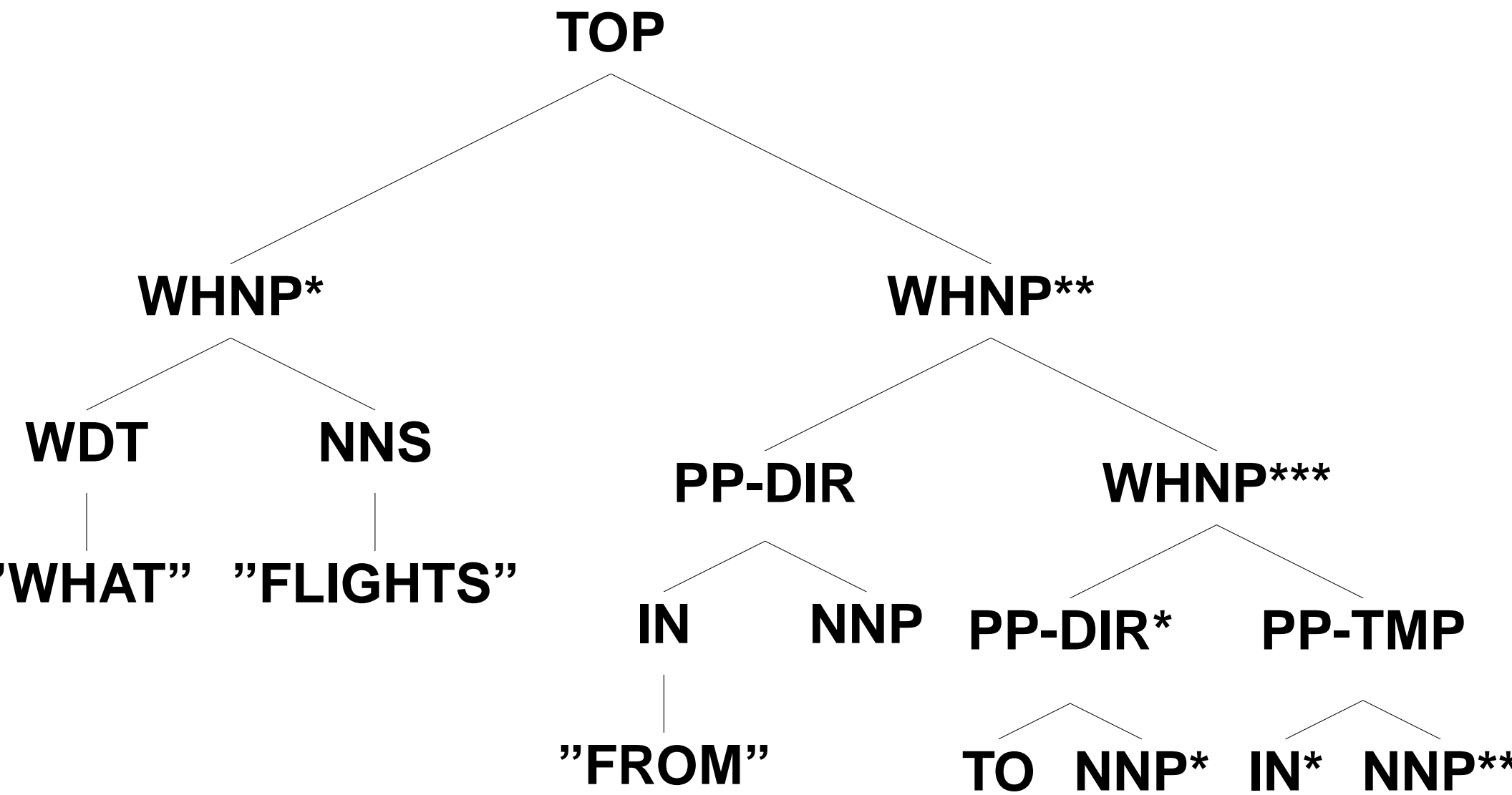
8.80 0.078 16



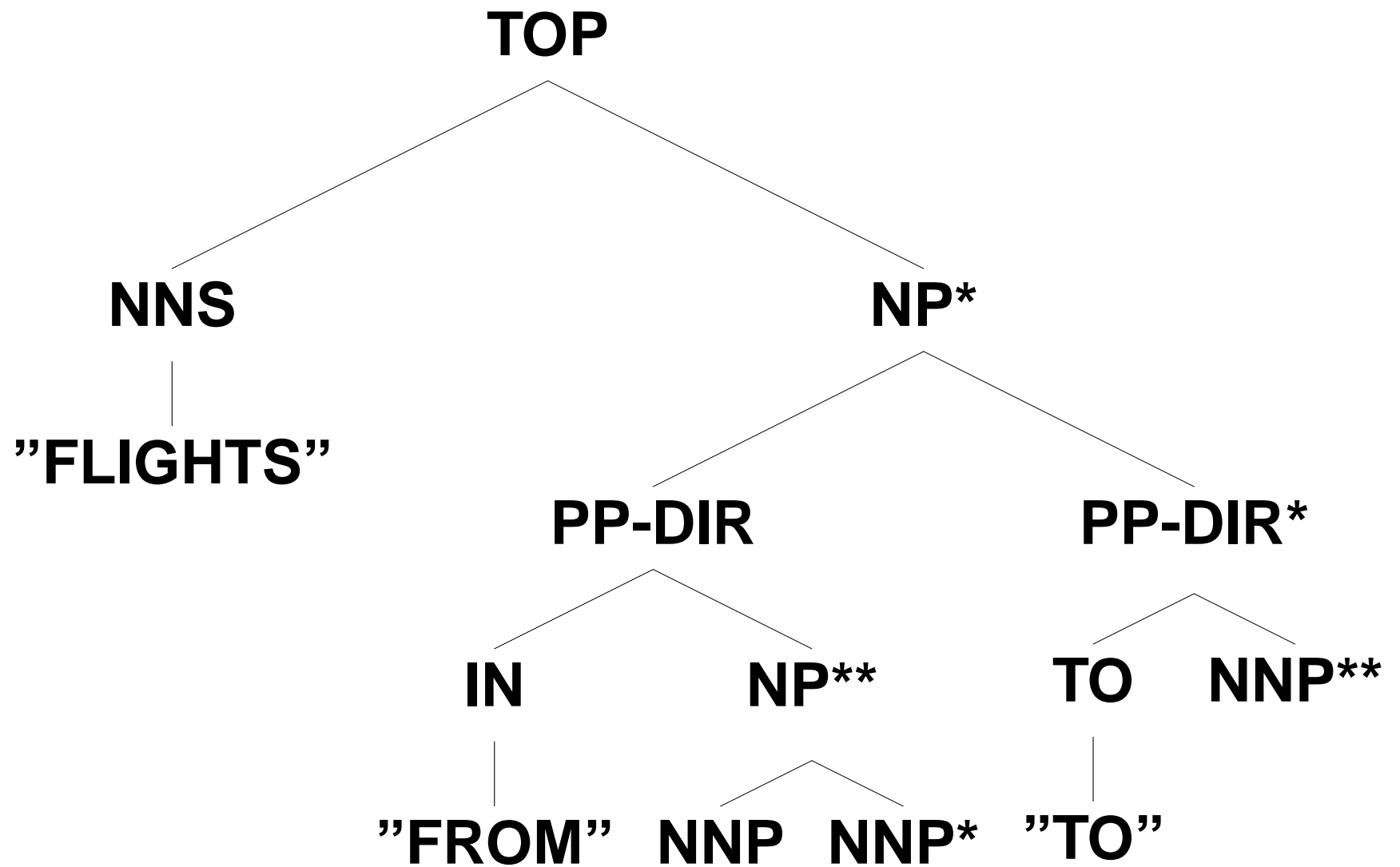
3.85 0.003 14



3.30 0.002 26



2.28 0.001 12



2.20 0.001 10

Conclusions

- Current theories of language hold that language users maintain a large, redundant and heterogeneous storage of constructions;
- If this view is correct, we expect to see statistical signatures of such constructions in corpora;
- We have presented a novel approach to identifying candidate constructions, based on:
 - learning an STSG from a tree bank
 - * using an new estimator “push-n-pull”
 - an expression for expected occurrence frequencies.
- Results on a small corpus suggest that the new method is able to learn STSGs that do well in parsing novel sentences, and make linguistically relevant generalizations.

Future work

- Bigger, faster, better...

Thanks to: NWO 612.066.405, Yoav Seginer, Rens Bod, Remko Scha, 3 anonymous reviewers and you all!

Subtree scores

$$\mathbf{E}[f(t)] = \sum_{d \in D(t)} (\alpha(d)\beta(d))$$

$$\alpha(d) = \sum_{\tau \in \widehat{tw}(d_1)} \left(\sum_{\tau' \in \widehat{pr}_{x(t)}(\tau)} u(\tau') \right) = \alpha_{d_1, x(t)}$$

$$\begin{aligned} \beta(d) &= \prod_{\substack{t' \in \\ \langle d_2, \dots, d_n \rangle}} \left(\sum_{\tau' \in \widehat{pr}_{x(t)}(t')} w(\tau') \right) \\ &= \prod_{\substack{t' \in \\ \langle d_2, \dots, d_n \rangle}} \frac{1}{\sum_{t'' : r(t') = r(t'')} u(t'')} \underbrace{\left(\sum_{\tau' \in \widehat{pr}_{x(t)}(t')} u(\tau') \right)}_{\beta_{t', x(t)}^*} \end{aligned}$$

With every change of the scores of t , keep track of the “subtree scores” of its supertrees.

update-scores(τ, δ)

$$u(\tau)_+ = \delta$$

for each prune τ' at sites x

$$\beta_{\tau',x}^* + = \delta$$

for each twig τ'' of τ

$$\alpha_{\tau'',x} + = \delta$$